

I hereby certify that this paper is being deposited with the United States Postal Service as Express Mail in an envelope addressed to: BOX PATENT APPLICATION, ASSISTANT COMMISSIONER FOR PATENTS, Washington, D.C. 20231, on this date.

March 30, 2001

Date

Express Mail No. **EL846166136US**

Inventors: David Kumpf; Armando Guzman

## A METHOD AND SYSTEM FOR ASSIGNING PERIPHERAL DEVICES TO LOGICAL PORTS OF A NETWORK PERIPHERAL SERVER

The present invention relates to computer servers, and particularly to a system and method for connecting multiple peripheral devices to a peripheral server, principally in a network environment.

### BACKGROUND OF THE INVENTION

Peripheral server products, such as the Hewlett-Packard JetDirect EX, exist today that perform the function of connecting peripherals in a network environment, wherein the peripherals use a parallel port interface (IEEE 1284). For example, print servers provide a well-known and widely used network interface for network client computers to send print jobs to a printer connected with a parallel cable to a given port. In such devices, print jobs are routed to the printer connected to the given port, for example port 1, no matter what printer is currently connected there. Much software exists that uses this model, including software for driving print spoolers.

A problem can arise when a different printer is connected to the same port that may be a different type of printer having the different capabilities compared to the

1 prior printer. Because the new printer necessarily receives any jobs send to that port, the  
2 job may not print correctly. This problem becomes particularly acute when the server  
3 and peripherals use the USB (Universal Serial Bus) interface. With the USB interface, all  
4 peripherals on the bus are assigned addresses randomly every time they are turned on.  
5 Thus, it becomes possible for two peripherals to exchange addresses even when neither  
6 has been disconnected. Also, typical USB servers have just one port and use intermediate  
7 hubs to connect multiple peripherals.

8           The different ports that client software communicates with can be thought  
9 of as “logical ports”. The physical connectors that the cables from the peripherals  
10 connect to can be thought of as “physical ports”. In existing parallel interface peripheral  
11 servers, even though there is a one-to-one correspondence between a logical port and a  
12 physical port, problems can occur as previously described. Furthermore, proper  
13 communication fails completely with USB when there are multiple peripherals connected  
14 to a single physical port.

15           While some of the problems may be alleviated by having the client  
16 software use a unique name for each printer instead of a logical port number, with the  
17 name being provided by the peripheral so it was the same, no matter where the peripheral  
18 was connected, the disadvantages of this solution include incompatibility with existing  
19 client software and the fact that many peripherals do not supply such a name.

20           Another alternative is for a USB server to have multiple physical ports,  
21 which then could be numbered sequentially and mapped one-to-one to logical ports.  
22 However, this solution does not solve the problem with moving printers and it requires a  
23 USB host controller for each physical port, which increases the cost of the product.  
24 Another alternative for USB devices is to use their USB address as their physical port  
25 number, which would again allow them to be numbered sequentially and mapped to  
26 logical ports. The disadvantage of this solution is that it also does not solve the problems  
27 with moving printers and randomly assigned USB addresses.

## 28 BRIEF SUMMARY OF THE INVENTION

1           The present invention is directed to a system and method that, among other  
 2 things, directs communications from client computers to peripheral devices using the  
 3 same sequentially numbered logical ports (port 1, port 2, etc.) that client software expects  
 4 to talk to, and assigns the same peripherals to the same logical ports every time they are  
 5 turned on. This is done substantially regardless of the type of physical port interface that  
 6 is being used to interconnect a peripheral server to the peripheral device.

7           The system and method, in its preferred embodiment, has a peripheral  
 8 server that automatically queries peripheral devices that are connected to a network for  
 9 identification information that is sent back and used by the server to assign a logical port  
 10 identification to respective peripheral devices. The identification information of the  
 11 peripheral devices, as well as the logical port identification and status of the logical port  
 12 is maintained in a table that has a predetermined maximum number of logical ports.

13           The system assigns logical ports that are free, and places peripheral devices  
 14 that have no logical port assigned in a queue until a logical port becomes free or a  
 15 reserved logical port is reassigned, according to specified criteria. The system can  
 16 operate automatically or in a manual assignment mode.

## 17           DESCRIPTION OF THE DRAWINGS

18           FIGURE 1 is block diagram of a network having a peripheral server and  
 19 client computers connected thereto, and having peripheral devices connected to the server  
 20 using parallel as well as USB connections, with the latter being made through a USB hub.

21           FIG. 2 is a logical port table having representative entries and status.

22           FIG. 3 is a flow chart of the operation of a preferred embodiment wherein a  
 23 new peripheral device is connected to the server.

24           FIG. 4 is a flow chart of the operation of a preferred embodiment after a  
 25 newly connected peripheral has been placed in a queue because of the unavailability of a  
 26 logical port.

## 27           DETAILED DESCRIPTION

28           In many network environments, and referring to the block diagram of FIG.  
 29 1, a peripheral server 10 provides a well-known and widely used network interface for

1 network client computers 12 to send peripheral function tasks via a network 14, such as  
2 print jobs, to a peripheral device 16 connected with a parallel cable 18 to a given port. In  
3 such devices, print jobs are routed to the printer connected to the given port, for example  
4 port 1, no matter what printer is currently connected there. The network environment  
5 may also include USB connected peripheral devices, which may be implemented by a  
6 port 20 connected to a USB hub 22 to which additional peripheral devices may be  
7 connected by USB cables 24.

8 In accordance with a preferred embodiment of the present invention, a  
9 system and method involves assigning logical ports for communicating with peripheral  
10 devices 16, regardless of the type of physical port interfaces that physically interconnect  
11 the peripheral device to the server 10. The system and method is adapted to be used with  
12 multiple peripheral devices being connected to a single server, but the actual number of  
13 possible logical port interfaces is preferably maintained at a maximum number that is  
14 likely to take care of the vast majority of installations. In this regard, it is believed that 4  
15 logical port entries is a reasonable number, although a larger number can certainly be  
16 used.

17 The system and method of the present invention is adapted for USB  
18 peripherals, parallel (IEEE 1284) peripherals, and all other known commercially  
19 available peripheral interfaces. A significant advantage of the present invention is that  
20 the operation of the peripheral devices by the client computer can be carried out without  
21 requiring any changes to the client software that is resident in the client computer.

22 In accordance with a preferred embodiment of the present invention, when  
23 the network peripheral server (e.g., a Hewlett-Packard JetDirect EX) is first initialized, it  
24 creates a table of the type shown in FIG. 2 that maps peripheral devices to logical port  
25 numbers. This table consists of a number of entries, i.e., preferably 4 as previously  
26 described, with each entry having at least the following information: logical port number,  
27 device identification, and current status.

28 The table is stored in some form of non-volatile memory, preferably in the  
29 server, so that previously mapped device information is available to the server every time

1 it reinitializes. Referring to the table in FIG. 2, examples of data are illustrated as shown,  
 2 including the logical port number, which is a number from one to the number of ports  
 3 supported, e.g., four. The device identification information in each entry preferably  
 4 uniquely identifies an individual device. In the preferred embodiment, the information  
 5 includes the device's manufacturer, model name, and a unique serial number. It should  
 6 be understood that additional or different information may be used. The current status  
 7 reflects whether this logical port is currently unused (FREE), has an attached peripheral  
 8 device currently assigned to it (ASSIGNED), or had a device assigned to a logical port in  
 9 the past that is not currently attached (RESERVED).

10 If a new peripheral device is attached after the server was initialized, the  
 11 preferred embodiment of the system and method carries out the process shown in the  
 12 flow chart of FIG. 3. After detection of the attachment (block 30), the server first  
 13 establishes communications with the device using well understood protocols such as  
 14 USB, IEEE-1284, etc. (block 32). Next, the server requests device identification  
 15 information (block 34). For USB devices, it issues standard USB requests for the  
 16 manufacturer, model, and serial number. For IEEE-1284 devices, it issues a standard  
 17 request for the device-id string, perhaps followed by vendor specific requests such as HP  
 18 PML (Printer Management Language) GET requests. The server combines all the device  
 19 information to form an identification key.

20 Next the server searches the table of previously assigned devices for a  
 21 matching key (block 36, 38). If an exact match is found, the server checks to determine if  
 22 the current status on that logical port is RESERVED (block 40). If RESERVED, it  
 23 indicates that it may have been the exact device that was previously connected or a  
 24 peripheral device that has the same capabilities as the previously assigned device and  
 25 should therefore be compatible, and the server assigns the device to the found logical port  
 26 (block 48). If the table entry indicates that the device is not RESERVED, then it must be  
 27 ASSIGNED, meaning there is a device with the same key already connected, enabled and  
 28 assigned to this logical port. In that unlikely event, the server searches the table for a  
 29 FREE entry (block 42) and if one is found (block 44), the server assigns the device's key

1 to the device identifier field (which was previously empty) in the table entry for the  
 2 selected logical port (block 46) and this peripheral device is then assigned to the found  
 3 logical port and the logical port activated so that client software can communicate with  
 4 the peripheral device (block 48). The status field of that logical port entry in the table is  
 5 then set to ASSIGNED (block 50), and the server returns to waiting for new devices  
 6 (block 30).

7 If there are no FREE entries, the server suspends processing this device for  
 8 some period to allow other devices that might just be initializing a chance to be assigned  
 9 to their RESERVED entries. This situation can arise, for example, when a cluster of  
 10 devices is powered on at the same time. The server adds the new device to a queue of  
 11 pending devices, starts a timer (block 52), and returns to waiting for new devices (block  
 12 30). The timer interval should be sufficient to allow other devices to complete their  
 13 initialization and establish communications with the server. For printer and multi-  
 14 function peripheral (MFP) devices, a good value is approximately two minutes, although  
 15 a greater or lesser value may be used.

16 In accordance with the preferred embodiment of the present invention, and  
 17 referring to FIG. 4, when the timer expires, the server removes the new device from the  
 18 pending queue and continues processing it (block 54). The server searches the table for a  
 19 RESERVED entry that provides a near match for the new peripheral device (block 56).  
 20 If multiple entries are found (block 58), the server picks the one that is the closest match  
 21 for the new device identification or key according to specified criteria by assigning the  
 22 device identifier to the found entry (block 60), assigns the assigned device to the found  
 23 logical port (block 62) and sets the status of it to ASSIGNED (block 64). For example,  
 24 an entry having the same manufacturer and model, but which differs only in the serial  
 25 number may be determined to be a closest match. There are also many other criteria that  
 26 could also be used to determine a closest match and can result in a very complex  
 27 implementation. The server may have a database containing information on many  
 28 peripheral devices, and may examine the database for other matching or similar criteria,  
 29 including similar functioning model numbers, for example. After the server replaces the

1 key with the new one and sets the current status to ASSIGNED, it returns to waiting for  
2 new devices (block 30).

3 If there are no RESERVED entries in the table (block 58), the server is at  
4 capacity. It logs the fact that the new device could not be added and therefore there is no  
5 overwrite (block 66), reports that the table is full (block 68) and returns to waiting for  
6 new devices. However, the system may permit a manual or automatic overwrite function  
7 to be carried out if it is desired. If an overwrite is to be done, the server searches the table  
8 for any entry with a RESERVED status (block 70), which if found (block 72) results in  
9 the assigning of the device identifier to the found entry (block 60) and the subsequent  
10 steps 62 and 64. If there were no RESERVED status entries found (block 72), the server  
11 report the table as being full.

12 During operation it is preferred that if a device is disconnected, the server  
13 will detect that condition and will locate the table entry for that device, set its current  
14 status to RESERVED, then return to waiting for new devices.

15 While the foregoing description covers the preferred embodiment which  
16 automatically assigns logical port. An alternative embodiment permits manual options to  
17 be available for users who prefer them. As noted in the background of the invention,  
18 existing servers that support IEEE-1284 parallel port devices automatically assign  
19 devices to logical ports based strictly on what parallel port they are connected to. For  
20 users or network administrators that wish to preserve this behavior, the server can offer  
21 an option to disable the new process and use the old one instead. As described above  
22 with respect to the preferred embodiment, the server assigns new devices to any free port  
23 and then re-assigns them to that same port every time they are reinitialized. Some users  
24 or network administrators may wish to manually assign new devices to a logical port of  
25 their choosing. In that case, the process described above can be replaced by one that puts  
26 all new devices in a pending queue. A user or network administrator can then assign  
27 devices via a user interface that presented a list of pending devices and a list of free ports.  
28 When the user selected a device and port combination, the server then assigns the device

1 to that logical port, stores the key in that table entry, and sets the current status to  
2 ASSIGNED.

3 From the foregoing, it should be appreciated that a system and method of  
4 assigning logical port numbers to peripheral devices that provides greater convenience  
5 and reliability in the operation of peripheral devices in a network environment where the  
6 peripheral devices may be physically connected to a peripheral server by a number of  
7 different physical port interfaces. The system is adapted to automatically assign logical  
8 port numbers and also reassign them as necessary in accordance with specified criteria.  
9 The system and method can be easily implemented at a relatively low cost.

10 While various embodiments of the present invention have been shown and  
11 described, it should be understood that other modifications, substitutions and alternatives  
12 are apparent to one of ordinary skill in the art. Such modifications, substitutions and  
13 alternatives can be made without departing from the spirit and scope of the invention,  
14 which should be determined from the appended claims.

15 Various features of the invention are set forth in the appended claims.  
16